PATENT ABSTRACTS OF JAPAN

(11)Publication number : 04-010051
(43)Date of publication of application : 14.01.1992

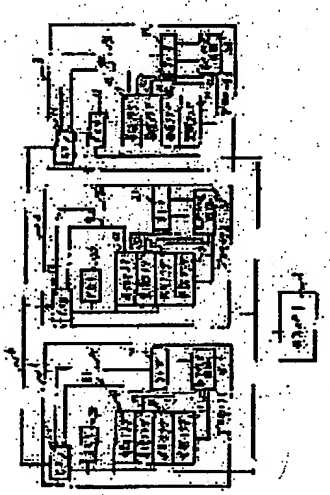
(51)Int.Cl. G06F 15/16
G06F 12/08

(21)Application number : 02-110003
(22)Date of filing : 27.04.1990

(71)Applicant : HITACHI LTD
(72)Inventor : YOSHIDA TOSHIBUMI
YABUSHITA MASAHARU

(54) SHARED MEMORY CONTROL METHOD FOR MULTIPROCESSOR SYSTEM

(57)Abstract:
PURPOSE: To attain the control of a suitable shared memory by providing a timer in each processor and updating the shared data stored in the processor if the data copied to an individual memory from the shared memory are not rewritten even after a prescribed time.
CONSTITUTION: Each of processors 11 - 13 rewrites and updates simultaneously the corresponding shared data blocks of all processors except its own processor or reads the data block outputted to a communication channel from a single processor into its own shared data block and updates the data block. At the same time, the timers 31 - 33 count the time passed after each shared data block is rewritten. When the counted time exceeds the prescribed time, this fact is informed to its own processor. Then the shared data blocks of all processors corresponding to the data blocks that exceeded the prescribed time are updated. Thus it is possible to attain the control of a shared memory suited to the updating of the shared data blocks in a multiprocessor system.



LEGAL STATUS
[Date of request for examination]
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's

⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平4−10051

⑤Int.Cl.⁵	識別記号	庁内整理番号	㊸公開　平成4年(1992)1月14日
G 06 F 15/16	350 R	8840−5L	
12/08	310 C	7232−5B	

審査請求　未請求　請求項の数 6　(全11頁)

㊹発明の名称　マルチプロセッサシステムの共有メモリ制御方法

㉑特　願　平2−110003
㉒出　願　平2(1990)4月27日

㉗発明者　吉田　俊文　神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
㉗発明者　薮下　正治　神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
㉑出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地
㉔代理人　弁理士　小川　勝男　外1名

明　細　書

1．発明の名称

マルチプロセッサシステムの共有メモリ制御方法

2．特許請求の範囲

1．複数のプロセッサと共有メモリが同一通信路上に接続され、前記各プロセッサが個別のメモリを持ち、前記各プロセッサが前記共有メモリの一部データを該個別のメモリにそれぞれコピーし、前記各プロセッサが前記個別のメモリにコピーした該データを並行してアクセスして処理を行うマルチプロセッサシステムにおいて、前記各プロセッサにタイマを設け、前記共有メモリから前記個別のメモリにコピーした前記データを前記プロセッサが書き換えることにより、該プロセッサの前記タイマが始動し、その後該プロセッサが該データに対する書き換えを該タイマの規定する時間内に行う度に、該タイマの測定する時間を初期値に戻し、そして、該プロセッサが該データに対する書き換えを前記タイマの規定する時間以上経過しても行わなかったとき、該タイマが停止すると同時に、該プロセッサが書換えた前記データを、前記データに対応する識別を有する他のプロセッサにブロードキャストすることによって、前記他プロセッサ内の共有データを更新することを特徴とする共有メモリ制御方法。

2．複数のプロセッサと共有メモリが同一通信路上に接続され、前記各プロセッサが個別のメモリを有し、前記各プロセッサが前記共有メモリの一部データを該個別のメモリにそれぞれコピーし、前記各プロセッサが前記各個別のレモリにコピーした該データを並行してアクセスして処理を行うマルチプロセッサシステムにおいて、前記各プロセッサにタイマを設け、前記共有メモリから前記個別のメモリにコピーした前記データを前記プロセッサが書き換えることにより、該プロセッサの前記タイマが始動し、その後、該タイマの規定する時間になると、該タイマが停止すると同時に、該プロセッサが書換えた前

記データを、前記データに対応する複製を有する他のプロセッサにブロードキャストすることによって、前記他プロセッサ内に共有データを更新することを特徴とする共有メモリ制御方法。

3．複数のプロセッサと共有メモリが同一通信路上に接続され、前記各プロセッサが個別のメモリを有し、前記各プロセッサが前記共有メモリの一部データを該個別のメモリにそれぞれコピーし、前記各プロセッサが前記各個別のメモリにコピーした該データを並行してアクセスして処理を行うマルチプロセッサシステムにおいて、前記各プロセッサにカウンタを設け、前記プロセッサの該カウンタが、前記共有メモリから前記個別のメモリにコピーした前記データを該プロセッサが書き換える回数をカウントし、該書き換えた回数が前記カウンタの規定する回数以上に達したとき、該プロセッサが書換えた前記データを、前記データに対応する複製を有する他のプロセッサにブロードキャストすることによって、前記他プロセッサ内の共有データを更新することを特徴とする共有メモリ制御方法。

4．複数のプロセッサが同一通信路上に接続され、前記各プロセッサが個別のメモリを有し、前記プロセッサが該個別のメモリに前記別プロセッサの個別メモリの一部データをコピーし、前記プロセッサが個別のメモリにコピーした該データをアクセスして処理を行うマルチプロセッサシステムにおいて、前記各プロセッサにタイマあるいはカウンタを設け、前記プロセッサがコピーした該データを書き換えた後、前記タイマまたはカウンタの規定する条件を満たしたとき、前記プロセッサが書換えた該データを、該データに対応する複製を有する他のプロセッサにブロードキャストすることによって、前記他プロセッサ内の共有データを更新することを特徴とする共有メモリ制御方法。

5．前記各プロセッサの前記タイマにレジスタを設け、該レジスタの保持する値が、前記タイマの規定する時間となり、該レジスタ値を前記各プロセッサが随時変更できることを特徴とする特許請求の範囲第1項あるいは第2項あるいは第4項のマルチプロセッサシステムの共有メモリ制御方法。

6．前記各プロセッサの前記カウンタにレジスタを設け、該レジスタに保持する値が、前記カウンタの規定する書き換え回数となり、前記プロセッサの個別のメモリに保持するデータ別に前記カウンタと該記レジスタを用意し、前記データ別に前記カウンタの規定する書き換え回数を前記プロセッサが随時設定できることを特徴とする特許請求の範囲第3項あるいは第4項のマルチプロセッサシステムの共有メモリ制御方法。

3．発明の詳細な説明

〔産業上の利用分野〕

本発明は、マルチプロセッサシステムにおける共有メモリの制御方法に関する。

〔従来の技術〕

現在のマルチプロセッサシステムにおいては、各プロセッサによる共有メモリ内データへのアクセスを速くするためと通信路（バスまたはネットワーク）の競合を少なくするために、各プロセッサが共有メモリの一部データ（以後、データブロックと呼ぶ）を各個別のメモリ（キャッシュ・メモリまたはメイン・メモリ）にコピーし、そのコピーしたデータブロックをアクセスすることで、間接的に共有メモリの各データブロックをアクセスする方式を用いる。そして、その各プロセッサは共有メモリの同一データブロックから（各個別のメモリに）コピーしたデータブロック（以後、共有データブロックと呼ぶ）の一貫性を保ちながら並行して処理を進める。

従来例としては、ジェームズ　アーチボルド（James Archibald）他著、「キャッシュ　コヒーレンス　プロトコルズ（Chache Coherence protocols）」に密結合マルチプロセッサシステムにおいてのキャッシュ・メモリの一貫性制御方式が紹介されている。

この文献で紹介されているキャッシュ・メモリの制御方式では、共有データブロックを他プロセッサが持つことや、データブロックの内容が自プ

ロセッサによって書き換えられて共有メモリのコピー元のデータブロックの内容と異なることや、そのデータブロックの内容が有効であること等のデータブロックの状態を示すもの（以後、フラグと呼ぶ）を用いる。各プロセッサは、それらフラグの状態により、各プロセッサの個別のメモリに保持する各データブロックの状態を認識し、そのデータブロックの状態に従ったその他プロセッサ及び共有メモリのデータブロックの必要最小限の更新を行う。

また、疎結合システムにおいても以上と同様なデータブロックの状態を管理して制御を行う方式を採用するものがある。一般に、データブロックの状態によって共有メモリの一貫性を制御するそれら方式では、複数のプロセッサが共有メモリの同一データブロックを各個別のメモリにコピーし、あるプロセッサがそのデータブロックの内容を書き換えると、その他全てのプロセッサの対応する共有データブロックの内容が無効となる。そして、あるプロセッサがその書き換えられた最新のデータブロックを必要としたとき、その書き換えられたデータブロックの内容を、自プロセッサの共有データブロックに書き込み更新する。

〔発明が解決しようとする課題〕

マルチプロセッサシステムにおいて、一般に、各プロセッサが共有データブロックを書き換える（または、アクセスする）頻度が多いと、各プロセッサ間と共有メモリを接続する通信路の競合が大きくなり、各プロセッサの通信路の利用待ち状態が長くなり、各プロセッサの処理能力が低下する。以上通信路の競合をできるかぎり避けようとする制御方式として、前記従来例のフラグを用いてデータブロックの状態を制御する方式が用いられる。

しかし、各プロセッサが共有データブロックを書き換える頻度が少ない場合、従来例の方式のように各プロセッサの要求時に１データブロックずつ更新していく方法は効率が悪い。そして、複数プロセッサの共有データブロックを更新するのに要する処理時間と１プロセッサの共有データブロックを更新するのに要する処理時間がほぼ等しい場合、共有データブロックが書き換えられると同時に他プロセッサの対応する各共有データブロックも書き換え更新する方が通信路の競合が少なくなり、また各プロセッサが即時に更新の共有データブロックをアクセスすることができる可能性が高くなる（勿論、２プロセッサ間で共有しているデータブロックの場合は除く）。

しかし、以上の書換えと同時に更新する方法では、各プロセッサの共有データブロックへの書き換えが一時期に集中してしまった場合に不利である。そして、その書換えが集中する間、以上の更新に関係する全てのプロセッサの処理が何度も中断され、また、どのプロセッサも参照していないのに次の更新が行われたりして、逆に効率が悪くなりまたオーバーヘッドが大きくなってしまう。

本発明の目的は、マルチプロセッサシステムの上述した共有データブロックを更新するのに好適な共有メモリ制御方法を提供することにある。

〔課題を解決するための手段〕

前記目的を実現する手段として、第一にシステムは、各プロセッサが自プロセッサ以外のその他全てのプロセッサの対応する共有データブロックを同時に書き換え更新するか、または１プロセッサが通信路へ出力するデータブロックをその他全てのプロセッサが並行して自共有データブロックに読み込み更新する機構を持つ。第二に各プロセッサは、それぞれ以下のタイマを持つ。そのタイマは、自プロセッサの各共有データブロックが（各プロセッサによって）書き換えられてからの時間を測定する。そして、その共有データブロックが書換えられる度にタイマのそのデータブロックに対する測定時間を０に戻し、もしその測定時間がタイマの規定する時間以上に達した（書き換えがその規定時間内で行われなかった）とき、自プロセッサにその事を知らせる機構を持つ。第三に各プロセッサは、第二の機構の上記タイマからの知らせを受け取り、第一の機構を用いて、その書き換えられてから規定時間以上経過したデータブロックに対応するその他全てのプロセッサの共

有データブロックを更新する機構を持つ。以上の3つの機構により、書換えが一時期に纏めて行われ、参照に対して書換える頻度が少ない（書換えられる周期が、参照される周期より長い）ような共有データブロックの最適な更新を実現する。

〔作用〕

前記第一の機構により、複数プロセッサの共有データブロックを更新する処理時間と、1プロセッサの共在データブロックを更新する処理時間とが殆ど等しくなる。前記第二の機構により、自プロセッサの各データブロックに対する書換えの集中する時期の終わりを認識し、その書換時期の終わりを自プロセッサに知らせることができる。前記第三の機構により、第一の機構と第二の機構を結び付け、その他全てのプロセッサの共有データブロックに更新を制御する。

以上3つの機構により、マルチプロセッサシステムにおいて、プロセッサが共有データブロックを一時期に纏めて書換えた後、その纏めて書換えたデータブロックをその他全てのプロセッサの対応する共有データブロックに書き込み、それら共有データブロックを一度に更新するという本発明の共有メモリ制御方法を実現する。その他、本発明の方法において、各プロセッサの共有データブロック毎に、本発明の以上3つの機構を用いるか否かを示すフラグ（以後、更新フラグと呼ぶ）を設け、適用するデータブロックを選択できるようにすることが考えられる。

またタイマを用いた別の実現方法として、前記第二の機構でデータブロックを書き換える度にタイマの測定する時間を0に戻すのを止め、最初に書き換えてから一定時間経つと（その間何度か書き換えられるかもしれないが）その書換えたデータブロックをその他全てのプロセッサの共有データブロックに書き込み更新する方法が考えられる。この方法の場合は、纏めて書き換えが行われる時間が予め分かっている場合に有効である。また、タイマを用いる方法では、各プロセッサの処理状況および各プロセッサの共有データブロックの内容（更新を要する頻度）によって、各プロセッサが各プロセッサのタイマの規定時間を変えられるようにすることが考えられる。

タイマ以外を用いて実現する方法として、前記第二の機構のタイマの代わりに複数個のカウンタを設ける。そして、それらカウンタは、自内部メモリに保持する各共有データブロックを自プロセッサが書き換える回数をカウントし、その回数がカウンタの規定する回数以上になったとき、その規定回数以上書換えたデータブロックをその他全てのプロセッサの共有データブロックに書き込み更新する方法が考えられる。この方法の場合は、一度に纏めて書き換える回数が予め分かっているデータブロックに対して有効である。また、このカウンタを用いる方法において、各共有データブロックの規定書き換え回数を、各プロセッサが随時変えられるようにすることが考えられる。

〔実施例〕

以下、本発明の実施例を図面により説明する。

第1図は、本発明の一実施例のマルチプロセッサシステムのブロック図である。3台のプロセッサ1，2，3と共有メモリ4がバス5に接続されていて、各プロセッサ1，2，3は、その他のプロセッサ1，2，3に対して割り込み信号6を出力することができる。例えば、プロセッサ1はプロセッサ2，3に対して割り込み信号6を同時に出力することができる。

各プロセッサ1，2，3は、内部にCPU11，12，13とメモリ21，22，23とタイマ31，32，33と4種のフラグ4i，5i，6i，7i（i＝1，2，3）と条件判定回路151，152，153を有する。各プロセッサ1，2，3は、アクセスする共有メモリ4の一部データブロック（データブロックのアドレスと内容）を内部メモリ21，22，23にコピーし、そのコピーしたデータブロックをアクセスして処理を行う。その各データブロックの内容は、以上のシステムにおいて予め定められた一定のデータサイズである。そして、その各データの（コピーした）アドレスは、共有メモリ4上のアドレスである。各プロセッサ1，2，3は、プロセッサ番

号とそのアドレスを含むシステムアドレスにより、その他プロセッサ１，２，３上の対応する共有データブロックをアクセスする（プロセッサ番号１，２，３は、それぞれプロセッサ１，２，３を指す）。

また特に、対応するプロセッサ１，２，３は存在しないが仮に定めたプロセッサ番号(xとする)とそのデータブロックのアドレスを含む特別なシステムアドレスによって、自プロセッサ１，２，３以外のその仕全てのプロセッサ１，２，３の対応する共有データブロックの内容をブロードキャストによって同時に書き換えることができる。例えば、プロセッサ１は、その特別なシステムアドレスを用いてプロセッサ２，３の共有データブロックを同時に書き換えることができる。各プロセッサ１，２，３の内部フラグ４ｉ，５ｉ，６ｉ，７ｉ（ｉ＝１，２，３）は、そのデータブロックの内容の有効性を示す有効フラグ４１，４２，４３と、そのデータブロックの内容を自内部ＣＰＵ１１，１２，１３が書換えたこと（および、各プロセッサ１，２，３の中で唯一有効状態であること）を示す書換フラグ５１，５２，５３と、そのデータブロックを他プロセッサ１，２，３と共有していることを示す共有フラグ６１，６２，６３と、条件が満たされたとき対応する共有データブロックを（ブロードキャストによって）一度に更新することを示す更新フラグ７１，７２，７３を有する。各プロセッサ１，２，３は、内部メモリ２１，２２，２３にコピーした各データブロックの状態を以上４つのフラ４ｉ，５ｉ，６ｉ，７ｉ（ｉ＝１，２，３）の値で管理する。

各プロセッサ１，２，３が処理に必要な共有メモリ４のデータブロックを自内部メモリ２１，２２，２３にコピーすると、以上の４つのフラグ４ｉ，５ｉ，６ｉ，７ｉ（ｉ＝１，２，３）が決定される。例えば、プロセッサ１が共有メモリ４のあるデータブロックを自内部メモリ２１にコピーしたとする。そのとき、その有効フラグ４１は有効状態を示し、その書換フラグ５１は書き換えられていない状態を示す。その共有フラグ６１は、すでに共有メモリ４のそのデータブロックを内部メモリ２２，２３にコピーしているプロセッサ２，３がいると共有状態を示し、そうでなければ固有状態を示す。更新フラグ７１は、コピーしたデータブロックの（共有メモリ４上の）アドレスにより決定される。（共有メモリ４の予め決まった領域に、ブロードキャストを用いた一度に更新するのに適したデータブロックが格納される。）

また、各プロセッサ１，２，３の処理や内部メモリ２１，２２，２３の記憶容量の都合により、プロセッサ１，２，３からデータブロックが追い出される。その場合、その追い出されるデータブロックの有効フラグ４１，４２，４３が有効状態に示し、書換フラグ５１，５２，５３が書き換えられた状態を示しているとき、そのデータブロックの内容は共有メモリ４のコピー元に書き戻される。そして、その他のデータブロックの状態の場合と同様に、そのデータブロックに対応する４つフラグ４ｉ，５ｉ，６ｉ，７ｉ(ｉ＝１，２，３)が初期化された後、そのデータブロックは破棄される。

共有フラグ６１，６２，６３が固有状態を示している有効なデータブロックに対する参照と書換は、（データブロックの内容がコピーされてから最初に書き換えられたときに、その書換フラグ５１，５２，５３が書き換えられた状態に変わるが）、そのままそのデータブロックに対して行われる。以降では、共有フラグ６１，６２，６３が共有状態を示している共有データブロックに対する参照と書換について述べる。

第２図は、第１図の実施例のシステムにおける各プロセッサ１，２，３の共有データブロック参照時の処理手順を示している。例えば、プロセッサ１が、内部メモリ２１のある共有データブロックを参照しようとする。そのとき、その参照しようとするデータブロックに対応する有効フラグ４１の状態によって処理が分かれる（ステップ１００）。もし、その有効フラグ４１が有状態を示していれば、そのままそのデータブロックを参照する（ステップ１０２）。もし、その有効フラグ４１が無効状態（有効でない状態）を示してい

れば、そのプロセッサ1の内部CPU11に割り込み信号81が入る。それにより内部CPU11は、他のプロセッサ2，3に対して割り込み信号6を出力し、その内部メモリ22，23の対応する共有データブロックを捜索する。プロセッサ2，3は、割り込み信号6を受け取ると処理を中断してスリープ（何も処理しない）状態になる。内部CPU11は、プロセッサ2，3の共有データブロックのうち有効フラグ42，43が有効状態を示しているデータブロック（または、そのような共有データブロックがないときは、共有メモリ4の元データブロック）の内容を自内部メモリ21に読み込み更新する（ステップ101）。そして、再びプロセッサ2，3に割り込み信号6を出力した後、以上（ステップ101）で更新した自データブロックを参照する（ステップ102）。プロセッサ2，3はその割り込み信号6によってスリープ状態から解放され再び処理を継続する。

以上で例えば、プロセッサ1がプロセッサ2の共有データブロックの内容を読み込んだとする。

れば、そのプロセッサ1の内部CPU11に割り込み信号81が入る。それにより内部CPU11は、他のプロセッサ2，3に対して割り込み信号6を出力し（それにより、プロセッサ2，3は処理を中断してスリープ状態になり）、その内部メモリ22，23の対応する共有データブロックを捜索する。

内部CPU11は、プロセッサ2，3のそれら共有データブロックのうち有効フラグ42，43が有効状態を示しているデータブロック（または、そのような共有データブロックがないときは、共有メモリ4の元データブロック）の内容を自内部メモリ21に読み込み更新する（ステップ201）。そして、他プロセッサ2，3のそれら共有データブロックの有効フラグ42，43を無効状態にして（ステップ202）、プロセッサ2，3に割り込み信号6を再び出力した後、その更新したデータブロックの内容を書き換える（ステップ203）。その時、その有効フラグ41は有効状態を示し、書換え（ステップ203）後、その書換フラグ

その時のその共有データブロックの有効フラグ41，42は有効状態を示し、また書換フラグ51，52は書き換えられていない状態を示す。

第3図は、第1図の実施例のシステムにおける各プロセッサ1，2，3の共有データブロック書換時の処理手順を示している。例えば、プロセッサ1が、内部メモリ21のある共有データブロックを書き換えようとする。そのとき、その書き換えようとするデータブロックに対応する有効フラグ41の状態によって処理が分かれる（ステップ200）。

もし、その有効フラグ41が有効状態を示していれば、他プロセッサ2，3の対応する共有データブロックの有効フラグ42，43を無効状態にした（ステップ202）後、その更新したデータブロックの内容を書き換える（ステップ203）。それにより、その書換フラグ51は書き換えられた状態を示す。

ステップ200で、書き換えようとするデータブロックの有効フラグ41が無効状態を示してい

51は書き換えられた状態を示す。プロセッサ2，3はその再度の割り込み信号6によってスリープ状態から解放され処理を継続する。

第4図は、第1図の実施例のシステムにおけるタイマ31，32，33のブロック図であり、各プロセッサ1，2，3の内部メモリ21，22，23に格納できるデータブロック数が64個であった場合の例である。以下では、説明を簡単にするためプロセッサ1のタイマ31を例として取り上げる。タイマ31は、カウンタ600，601，…，663とレジスタ700と割り込み制御回路900を有する。各カウンタ600，601，…，663は、内部メモリ21の各データブロックに対応して存在する。例えば、カウンタ601は、内部メモリ21の（アドレス0から格納されている順に0から数えて）第1番目のデータブロックに対応する。

そして、各カウンタ600，601，…，663は、入力されるクロック信号300をカウントする。カウンタ600，601，…，663のその

他入力信号としては、カウントを行う条件となる信号（以後、条件信号と呼ぶ）４００，４０１，…，４６３と、カウント数を０に戻す信号（以後、初期化信号と呼ぶ）５００，５０１，…，５６３がある。それら２種類の信号は、プロセッサ１の条件判定回路１５１から出力される。条件信号４００，４０１，…，４６３は、条件判定回路１５１から出力される。各データブロックに対応する４種のフラグ４１，５１，６１，７１の状態値の論理積をとった信号となる（それにより、データブロックが、共有状態にあり、内容が有効で、書き換えられており、書き換えられた後ブロードキャストによって対応する共有データブロックを一度に更新することを望んでいるという４つの条件のもとでカウントを行う）。

初期化信号５００，５０１，…，５６３は、各カウンタに対応するデータブロックの内容が内部ＣＰＵ１１によって書き換えられるとき、条件判定回路１５１から毎回出力される信号である（それにより、カウンタ６００，６０１，…，６６３は、対応するデータブロックが書き換えられる度にカウント数を初期化する）。また、タイマ３１の内部レジスタ７００は、各カウンタ６００，６０１，…，６６３のカウント数を規定する。各カウンタ６００，６０１，…，６６３は、そのカウント数がレジスタ７００の規定値以上になったとき、割り込み制御回路９００に更新を要求する信号８００，８０１，…，８６３を出力する（各カウンタ６００，６０１，…，６６３には、レジスタ７００の数値と比較をとる機能があるものとする）。

割り込み制御回路９００は、各カウンタ６００，６０１，…，６６３のうちどれかひとつでも更新を要求する信号８００，８０１，…，８６３を出力していると、更新用の割り込み信号９１を内部ＣＰＵ１１へ出力する。内部ＣＰＵ１１は、その割り込み信号９１を受け取るとタイマ３１の割り込み制御回路９００の入力ポートを読み、どのカウンタ６００，６０１，…，６６３が更新を要求したか、延いてはどの共有データブロックを更新しなければならないかを認識する。そして内部ＣＰＵ１１は、その他プロセッサ２，３に割り込み信号６を出力し（それにより、プロセッサ２，３は処理を中断してスリープ状態になり）、認識した更新すべきデータブロックの内容を他プロセッサ２，３の対応する各共有データブロックに（ブロードキャストによって）同時に書き込み更新する。

その後、プロセッサ１は、再びプロセッサ２，３に割り込み信号６を出力する（それにより、プロセッサ２，３はスリープ状態から解放され処理を継続する）。以上の更新により、各プロセッサ１，２，３のその共有データブロックの有効フラグ４１，４２，４３は有効状態を示し、また書換フラグ５１，５２，５３は書き換えられてない状態を示す。以上、タイマ３１とタイマ３２，３３は等しく、条件判定回路１５１と条件判定回路１５２，１５３は等しいものとする。

第５図は、第４図のタイマ３１，３２，３３の各カウンタ６００，６０１，…，６６３における

クロック信号３００が入力された時の動作フローである。例えば、カウンタ６０１は、クロック信号３００が入力されたとき（例えばクロック信号３００が立ち下がるとき）、条件信号４０１の入力値によりカウントするかどうかが決まる（ステップ１０００）。条件信号４０１が１であれば、カウント数をカウントアップし（ステップ1001）、もし、条件信号４０１が０であれば何もしない。そして、カウンタ６０１がカウント数をカウントアップした（ステップ１００１）場合、そのカウントアップしたカウント数が内部レジスタ７００の数値と比較される（ステップ１００２）。もし、カウント数をレジスタ７００の値以上であれば、割り込み制御回路９００に更新を要求する信号８０１を出力する（ステップ１００３）。

以上のタイマ３１，３２，３３と条件判定回路１５１，１５２，１５３を設けることにより、プロセッサ１，２，３の内部メモリ２１，２２，２３に保持する共有データブロック毎に、そのデータブロックが自内部ＣＰＵ１１，１２，１３に

より書き換えられてからの時間を測定することが可能となる（その測定時間は、対応するデータブロックが自内部ＣＰＵ１１，１２，１３により書き換えられる度に０に戻る）。また、ブロードキャストを用いて一度に更新を行うのに適当なデータブロックのみ書き換えられてからの時間を測定することができる。そして、タイマ３１，３２，３３のそのデータブロックに対する測定時間が、カウンタ内部レジスタ７００の規定値以上になるとき、割り込み信号９１，９２，９３を内部ＣＰＵ１１，１２，１３に出力し、内部ＣＰＵ11，１２，１３に対して、その書き換えられたデータブロックに対応するその他共有データブロックを更新することを要求することができる。

以下、第６図と第７図は、第１図の実施例のシステムの３台のプロセッサ１，２，３が共有するあるデータブロックに対するアクセスと、各プロセッサ１，２，３のその対応する共有データブロックの状態を示した例である。横軸を時間とし、縦矢印が各プロセッサ１，２，３のその共有データブロックに対するアクセスを表している。図の最初の横軸がプロセッサ１の共有データブロックに対するものであり、下２つの横軸がプロセッサ２とプロセッサ３の共有データブロックに対するものである。そして、プロセッサ１の横軸の下のもう一つ部分は、プロセッサ１のタイマ３１が測定するその共有データブロックに対する書換後の経過時間を表している。

第６図は、共有データブロックをブロードキャストによって一度に更新する最適な場合の例である。まずプロセッサ１は、そのデータブロックに対して、３回の連まった書換えを行う（ポイント２０００）。その第一の書換えで、プロセッサ２とプロセッサ３の共有データブロックは無効化され、プロセッサ１のタイマ３１は、そのデータブロックに対する書換え後の経過時間を測定し始める（ポイント２００１）。その後第２の書換えで、その測定時間を初期化され（ポイント２００２）、また、第３の書換えで、再び初期化される（ポイント２００３）。第３の書換え後、そのプロセッサ１のタイマ３１のそのデータブロックに対する測定時間は、そのタイマ３１の内部レジスタ700の規定時間に達する（ポイント２００４）。それにより、プロセッサ１は、そのデータブロックをプロセッサ２とプロセッサ３の対応する共有データブロックに書き込み、それら共有データブロックの内容を更新する（データブロックの内容が有効になる）。その後、プロセッサ２とプロセッサ３は、それら更新された共有データブロックを参照する。また、プロセッサ１は、その後２回連まった書換えを行い（ポイント２００５）、プロセッサ２とプロセッサ３の共有データブロックを再び無効化し（ポイント２００６）、そして、そのデータブロックに対するタイマ３１の測定値が規定時間に達したとき（ポイント２００７）、プロセッサ２とプロセッサ３の共有データブロックの内容を更新する。

第７図は、第６図において、プロセッサ１が最初の連まった書換え（ポイント３０００）を行っている途中に、プロセッサ２が対応する共有データブロックを参照しようとした場合の例である。第６図の場合と同様に、プロセッサ１による第一の書換え後、プロセッサ２とプロセッサ３の共有データブロックは無効化され、そして、同時にプロセッサ１のタイマ３１は、そのデータブロックに対する書換え後の経過時間を測定し始める（ポイント３００１）。その後、プロセッサ２の共有データブロックの参照により、タイマ３１は、そのデータブロックに対する測定を止める（ポイント３００２）。そして、そのプロセッサ１のその書き換えられたデータブロックを、プロセッサ２は自共有データブロックに読み込み更新し、そして、その更新したデータブロックを参照する。その後しばらくすると、プロセッサ１の第二の書換えが行われ、プロセッサ２の共有データブロックは再び無効化され、同時にプロセッサ１のタイマ３１は、そのデータブロックに対する書換え後の経過時間を再び測定し始める(ポイント３００３)。その後の状況は、各プロセッサ１，２，３とも第６図と同様である。

以上、本実施例のシステムでは、特別なシステムアドレスと各プロセッサ1，2，3相互の割り込み信号6により、共有データブロックの更新を実現する。また、共有データブロックの共有メモリ4上のアドレスにより、更新フラグ71，72，73の状態値を決定した。これは、処理される同一目的のデータが、共有メモリ4の決まったアドレス領域に格納されている場合の例である。この場合、本発明の共有データブロックをブロードキャストによって一度に更新する方法を適用するか否かを、各プロセッサ1，2，3が処理するタスクに適用することができる。（更新フラグ71，72，73が更新を示すアドレス領域に、本発明の方法に適したタスク部分を格納する。）また、各データブロック単位に、各データブロックの内容（テキスト，ローカル・データ，共有データの占める割合）に応じて共有メモリ4上の格納領域を変えることができれば、より適切な共有メモリ4の一貫性制御が行える。

また、各プロセッサ1，2，3が各タイマ31，32，33の内部レジスタ700の規定値を変えられるようにすると、各プロセッサ1，2，3の処理状態に適合したタイミングで共有データブロックの更新を行うことができる。

その他、第7図のプロセッサ1の書換が集中して行われている途中に（ステップ3000）他プロセッサ2でその共有データブロックに対する参照が行われたとき（ステップ3002），プロセッサ1のタイマ31を停止させる理由は，3台の内2台のプロセッサ1，2のすでに更新された共有データブロックが存在することになり、ブロードキャストによって一度に更新する効果が得れるからである。4台以上からなるマルチプロセッサシステムにおいては、以上でタイマ31，32，33が停止しないように各カウンタ600，601，…，663のカウントを行う条件（条件判定回路151，152，153）を変える必要がある。また、以上タイマ31，32，33のクロック信号300，条件信号400，401，…，463，初期化信号500，501，…，563を変えることにより、別のタイミングでデータブロックの更新を行うことができる。例えば、クロック信号300の代わりに他のイベント信号をカウントして更新を行うようなこともできる。

〔発明の効果〕

本発明は、マルチプロセッサシステムにおいては、各プロセッサが共有するデータブロックを一時期に纏めて書換えた後、その纏めて書換えたデータブロックをその他全てのプロセッサの対応する共有データブロックに書き込み、それら共有データブロックを更新する。そのため、書換えられる確率が低く（書換えが行われる間隔が長く）また一度に纏めて書換えが行われるというような性質を持つデータブロックおよび、各プロセッサがそのような共有データブロックに対するアクセス方法を採るマルチプロセッサシステムにおいて有利である。それに加えて本発明は、複数プロセッサのデータブロックを更新するのに要する時間と、1プロセッサのデータブロックを更新するのに要する時間が殆ど等しくなることを前提としている。それにより、多くのプロセッサによって共有される確率の高いデータブロックに対してや、各プロセッサが共有メモリの各データブロックを共有する確率が高いシステムにおいて有利である。

以上の有利な条件で、各プロセッサの要求時に1データブロックずつ更新するより、複数データブロックを一度に更新する本発明の方が、各プロセッサの処理のオーバヘッドおよび通信路の競合が少なくなる。また、あるプロセッサがある共有データブロックを書換えても、その後一定時間経つと、その他全てのプロセッサの対応する共有データブロックは一度に更新される。このことから、各プロセッサの共有データブロックの内容が最新のものである確率が高くなり、そして、すぐに（通信路を用いずに）アクセスできる確率が高くなる。本発明の方法を以上の条件にあった共有メモリのデータ更新に用いることにより、さらにマルチプロセッサシステムの処理を向上させることができる。

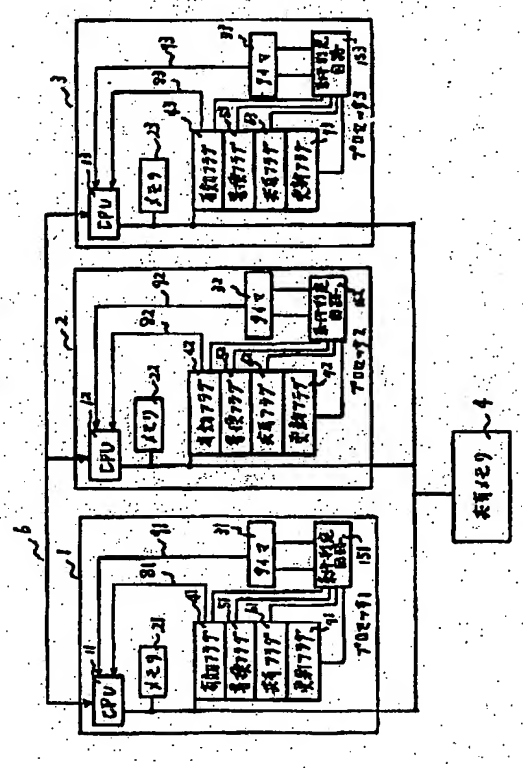
4．図面の簡単な説明

第１図は、本発明の実施例の３台のプロセッサによるマルチプロセッサシステムのブロック図である。第２図は、第１図の実施例のシステムにおける各プロセッサの共有データブロック参照時の処理手順図である。第３図は、第１図の実施例のシステムにおける各プロセッサの共有データブロック書換時の処理手順図である。第４図は、第１図の実施例のシステムにおける各プロセッサ内のタイマのブロック図である。第５図は、第４図のタイマ内部の各カウンタのクロック信号が入力された時の動作フロー図である。第６図と第７図は、第１図の実施例のシステムおける３台のプロセッサが共有するあるデータブロックに対するアクセスと、各プロセッサのその対応する共有データブロックの状態を表した説明図である。

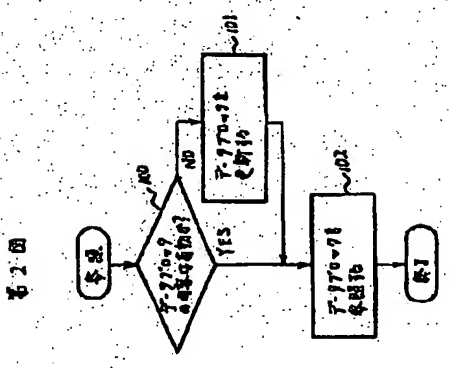
１，２，３…プロセッサ、４…共有メモリ、５…バス、６…ＣＰＵ外部割り込み信号、１１，１２，１３…ＣＰＵ、２１，２２，２３…メモリ、３１，３２，３３…タイマ、４１，４２，４３…有効フラグ、５１，５２，５３…書換フラグ、６１，６２，６３…共有フラグ、７１，７２，７３…更新フラグ、８１，８２，８３…ＣＰＵ割り込み信号（データ無効割り込み要求）、９１，９２，９３…ＣＰＵ割り込み信号（更新割り込み要求）、１５１，１５２，１５３…条件判定回路、３００…クロック信号、４００，４０１，…，４６３…カウンタ条件信号、５００，５０１，…，５６３…カウンタ・リセット信号、６００，６０１，…，６６３…カウンタ、７００…タイマ・レジスタ、８００，８０１，…，８６３…更新要求信号、９００…タイマ割り込み制御回路。

代理人　弁理士　小川勝男

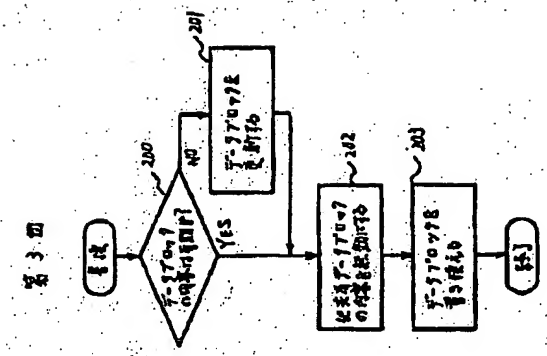
第１図



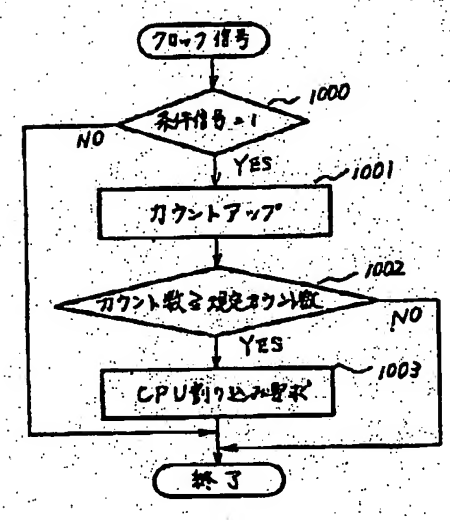
第２図



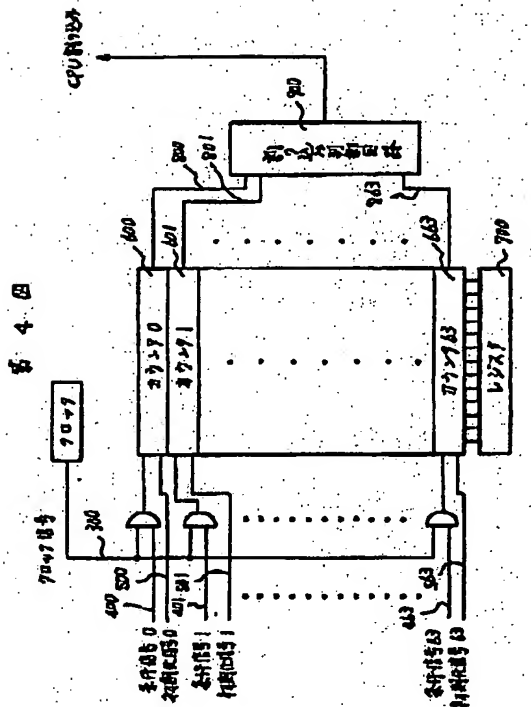
第３図



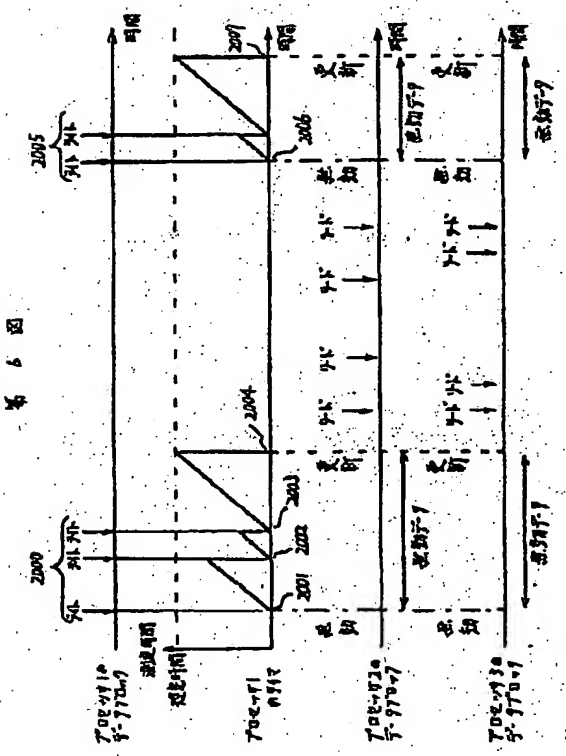
第5図



第4図



第6図



第7図